UNIVERSITY OF OXFORD

Machine Learning
Hilary Term 2016
Week 5

Problem Sheet 3

1 Bayesian View of Ridge and Lasso

Recall that the linear model for given parameter vector $\mathbf{w}$ and input $\mathbf{x}$ is defined as

$$y \sim \mathcal{N}(\mathbf{x}^T\mathbf{w}, \sigma^2)$$

As usual we will not model the inputs $\mathbf{x}_i$ as coming from some probability distribution, but consider them as fixed. Also, we will consider σ to be fixed and known. Thus, for data $\langle(\mathbf{x}_i, y_i)\rangle_{i=1}^m$ the likelihood given $\mathbf{w}$ (and the datapoints $\mathbf{x}_i$ and σ) can be expressed as

$$p(\mathbf{y} \mid \mathbf{w}; \mathbf{X}, \sigma) = \mathcal{N}(\mathbf{y} \mid \mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_m)$$

We will assume for this problem that the data, both inputs and outputs, has been centered. The notation $\mathcal{N}(\mathbf{y} \mid \mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_m)$ means that $\mathbf{y}$ is distributed according to the multivariate normal density $\mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_m)$. In class we studied two forms of regularization, the ridge and Lasso, with the following objective functions:

$$L_{\text{ridge}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w}^T\mathbf{w}$$
$$L_{\text{lasso}}(\mathbf{w}) = (\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\|\mathbf{w}\|_1$$

where $\|\mathbf{w}\|_1 = \sum_{i=1}^n |w_i|$.

1. In class we discussed how the estimate obtained from ridge regression can be viewed as the *maximum a posteriori* (MAP) estimate with a suitably chosen prior on $\mathbf{w}$, *i.e.*, $p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \tau^2\mathbf{I}_n)$. Make precise the relationship between σ, τ and λ for which the equivalence holds. Calculate the posterior distribution (without the normalization constant) and show that it takes the form of a multivariate Gaussian distribution. Thus in this case, you can actually explicitly compute the normalization constant of the posterior.

2. *Not required for submission*: Using the above posterior, for a new point $\mathbf{x}_{\text{new}}$ calculate the distribution $p(y_{\text{new}} \mid \mathbf{x}_{\text{new}}, \mathcal{D} = \langle(\mathbf{x}_i, y_i)\rangle_{i=1}^m)$ in the full Bayesian sense. This distribution is also a normal distribution with mean $\mathbf{x}_{\text{new}}^T\mathbf{w}_{\text{map}}$ and variance that dependes on $\mathbf{x}_{\text{new}}$. Explain why the variance depends on how different $\mathbf{x}_{\text{new}}$ is to the points in the dataset. (*Hint*: You will have to use properties of the eigendecomposition of the matrix that appears in the expression for variance.)

3. Describe how you would choose a suitable prior over $\mathbf{w}$ so as to view the Lasso estimate as a MAP estimate. Describe the relationship between λ, σ and any parameters you choose to represent the prior.

2 Optimization Methods for ℓ_1-regularization

1. Show that if you use the absolute loss function with the regularization term corresponding to Lasso (called ℓ_1 regularization as the penalty is on the ℓ_1 norm of the parameter vector), the optimization problem can be solved using linear programming. The objective function is:

$$L(\mathbf{w}) = \sum_{i=1}^{m} |\mathbf{x}_i^T \mathbf{w} - y_i| + \lambda \sum_{i=1}^{n} |w_i|$$

2. In case we use the squared loss, we can no longer use linear programming. Write the sub-gradient descent update rule with step size η, *i.e.,* write how you would obtain $\mathbf{w}_{t+1}$ using $\mathbf{w}_t$ and an (explicitly computed) subgradient of the objective function at $\mathbf{w}_t$ and step-size η. The objective function is:

$$L(\mathbf{w}) = \sum_{i=1}^{m} (\mathbf{x}_i^T \mathbf{w} - y_i)^2 + \lambda \sum_{i=1}^{n} |w_i|$$

3 Linear algebra revision: Eigen-decompositions

Once you start looking at raw data, one of the first things you notice is how redundant it often is. In images, it's often not necessary to keep track of the exact value of every pixel; in text, you don't always need the counts of every word. Correlations among variables also create redundancy. For example, if every time a gene, say A, is expressed another gene B is also expressed, then to build a tool that predicts patient recovery rate from gene expression data, it seems reasonable to remove either A or B. Most situations are not as clear-cut.

In this question, we'll look at eigenvalue methods for factoring and projecting data matrices (images, document collections, image collections), with an eye to one of the most common uses: Converting a high-dimensional data matrix to a lower-dimensional one, while minimizing the loss of information.

The Singular Value Decomposition (SVD) is a matrix factorization that has many applications in information retrieval, collaborative filtering, least-squares problems and image processing.

Let $\mathbf{X}$ be an $n \times n$ matrix of real numbers; that is $\mathbf{X} \in \mathbb{R}^{n \times n}$. Assume that $\mathbf{X}$ has n eigenvalue-eigenvector pairs $(\lambda_i, \mathbf{q}_i)$:

$$\mathbf{X}\mathbf{q}_i = \lambda_i \mathbf{q}_i \qquad i = 1, \ldots, n$$

If we place the eigenvalues $\lambda_i \in \mathbb{R}$ into a diagonal matrix $\mathbf{\Lambda}$ and gather the eigenvectors $\mathbf{q}_i \in \mathbb{R}^n$ into a matrix $\mathbf{Q}$, then the eigenvalue decomposition of $\mathbf{X}$ is given by

$$\mathbf{X} \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \ldots & \mathbf{q}_n \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \ldots & \mathbf{q}_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix} \tag{1}$$

or, equivalently,

$$\mathbf{X} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}.$$

For a symmetric matrix, i.e. $\mathbf{X} = \mathbf{X}^T$, one can show that $\mathbf{X} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$. But what if $\mathbf{X}$ is not a square matrix? Then the SVD comes to the rescue. Given $\mathbf{X} \in \mathbb{R}^{m \times n}$, the SVD of $\mathbf{X}$ is a factorization of the form

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

These matrices have some interesting properties:

- $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ is diagonal with positive entries (singular values σ in the diagonal).
- $\mathbf{U} \in \mathbb{R}^{m \times n}$ has orthonormal columns: $\mathbf{u}_i^T \mathbf{u}_j = 1$ only when $i = j$ and 0 otherwise.
- $\mathbf{V} \in \mathbb{R}^{n \times n}$ has orthonormal columns and rows. That is, $\mathbf{V}$ is an orthogonal matrix, so $\mathbf{V}^{-1} = \mathbf{V}^T$.

Often, $\mathbf{U}$ is m-by-m, not m-by-n. The extra columns are added by a process of orthogonalization. To ensure that dimensions still match, a block of zeros is added to $\mathbf{\Sigma}$. For our purposes, however, we will only consider the version where $\mathbf{U}$ is m-by-n, which is known as the *thin-SVD*.

It will turn out useful to introduce the vector notation:

$$\mathbf{X}\mathbf{v}_j = \sigma_j \mathbf{u}_j \qquad j = 1, 2, \ldots, n$$

where $\mathbf{u} \in \mathbb{R}^m$ are the left *singular vectors*, $\sigma \in [0, \infty)$ are the *singular values* and $\mathbf{v} \in \mathbb{R}^n$ are the right singular vectors. That is,

$$\mathbf{X} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_n \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{bmatrix} \tag{2}$$

or $\mathbf{X}\mathbf{V} = \mathbf{U}\mathbf{\Sigma}$. Note that there is no assumption that $m \geq n$ or that $\mathbf{X}$ has full rank. In addition, all diagonal elements of $\mathbf{\Sigma}$ are non-negative and in non-increasing order:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$$

where $p = \min(m, n)$.

Question: Outline a procedure for computing the SVD of a matrix $\mathbf{X}$. Hint: assume you can find the eigenvalue decompositions of the symmetric matrices $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}\mathbf{X}^T$.

4 Maximum Likelihood for Logistic Regression

Consider the sigmoid function, defined as $\sigma(z) = \frac{1}{1+e^{-z}}$. Note that $\lim_{z\to-\infty}\sigma(z) = 0$ and $\lim_{z\to\infty}\sigma(z) = 1$. Thus, for binary classification problems, we can compose a linear function with the sigmoid function to model the probability that a given input $\mathbf{x}$ belongs to one of the two classes $\{0, 1\}$. More precisely, for parameter vector $\mathbf{w} \in \mathbb{R}^n$, and input vector $\mathbf{x} \in \mathbb{R}^n$,[1] the label $y \in \{0, 1\}$ is given by the following model:

$$\Pr(y = 1 \mid \mathbf{w}, \mathbf{x}) = \sigma(\mathbf{x}^T\mathbf{w})$$
$$\Pr(y = 0 \mid \mathbf{w}, \mathbf{x}) = 1 - \sigma(\mathbf{x}^T\mathbf{w})$$

1. Show that the derivative of σ, $\sigma'(z) = \sigma(z)(1 - \sigma(z))$
2. Suppose you have data $\langle(\mathbf{x}_i, y_i)\rangle_{i=1}^m$. We consider the inputs $\mathbf{x}_i$ as fixed and only model the labels y_i as random variables. For model parameter $\mathbf{w}$ (and the fixed $\mathbf{x}_i$s) write the likelihood of observing the labels $y_1, \ldots y_m$. Note that each $y_i \in \{0, 1\}$.
3. Compute the gradient and Hessian of the negative log likelihood. Show that the Hessian is positive semi-definite.
4. What algorithm would you use to find the maximum likelihood estimate and what can you say about the obtained solution?

[1] As always we will assume that an extra dimension which takes value 1 on every data point has been added to avoid dealing with the constant term separately.